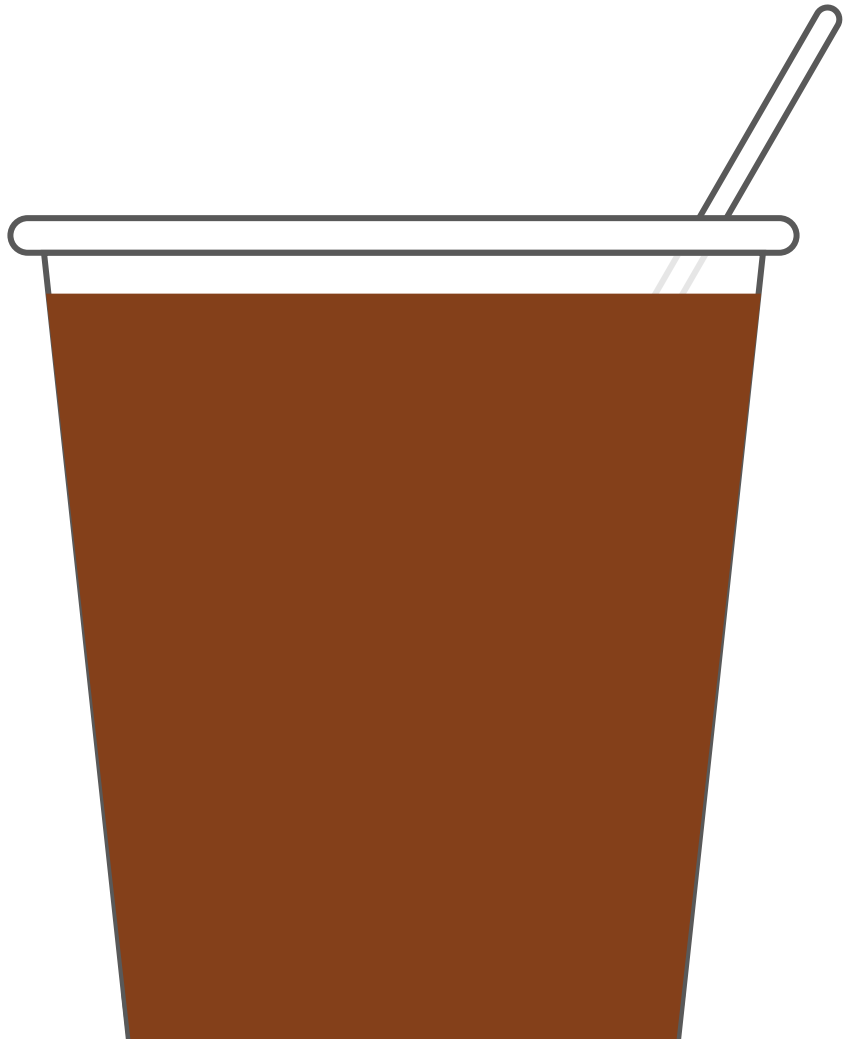
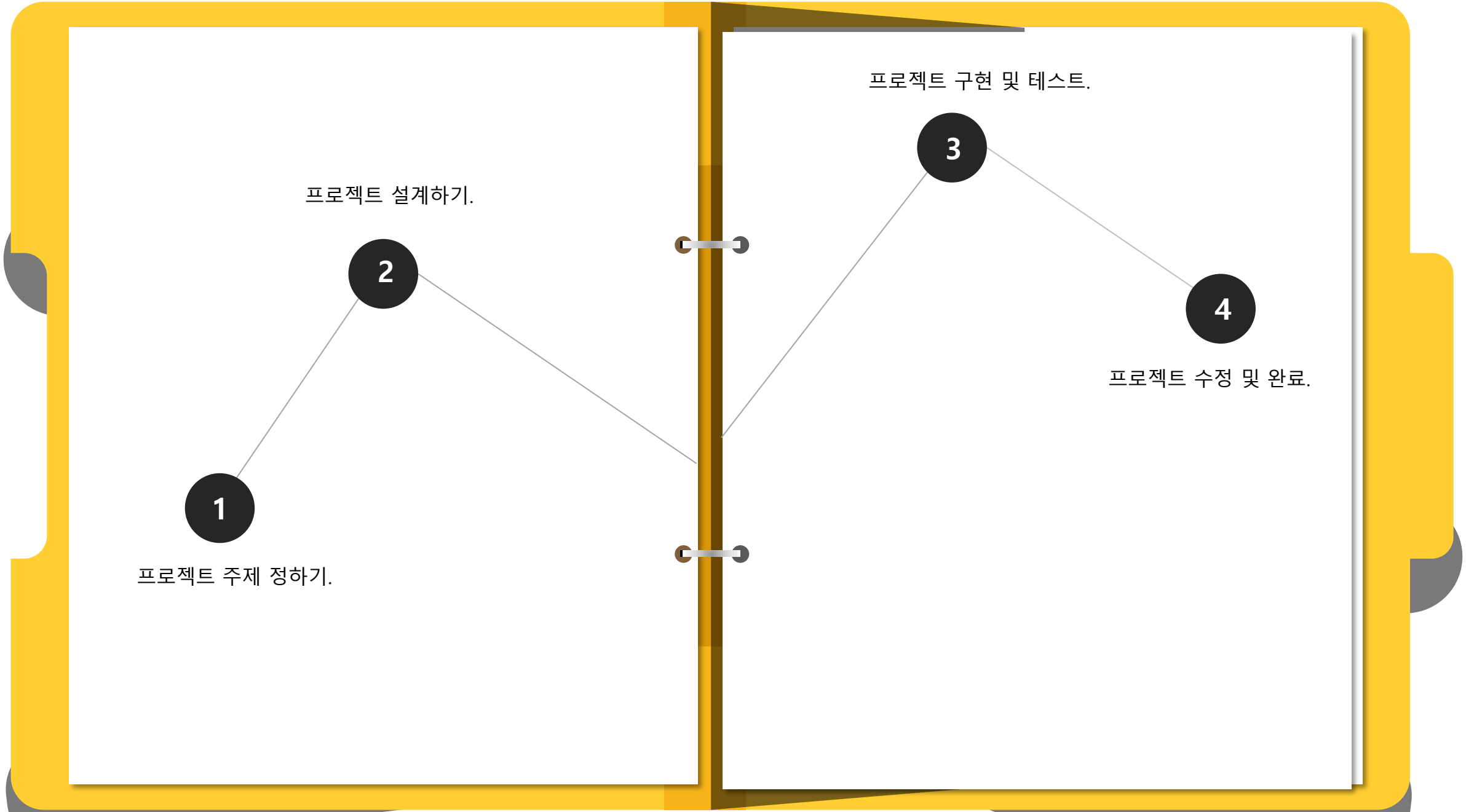
CAFÉ.BITLOCO;

노유리 , 유동준 , 소종원
김나연 , 최아리 , 송주은



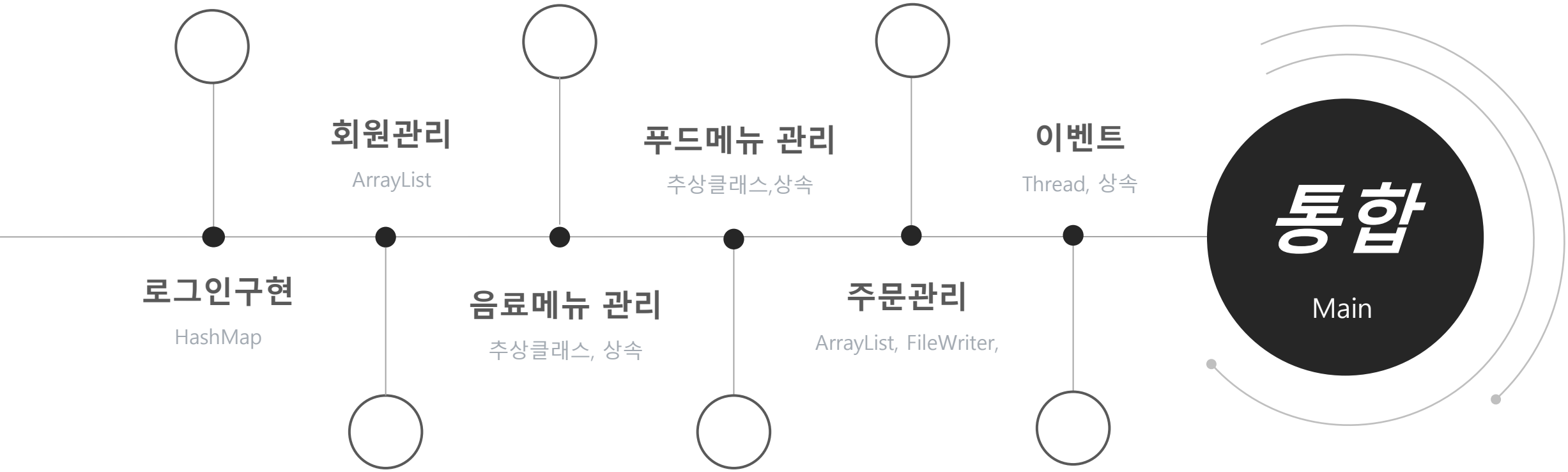
Project Architecture

Information Bitloco.


프로젝트 설계하기.
2
1
프로젝트 주제 정하기.
프로젝트 구현 및 테스트.
3
4
프로젝트 수정 및 완료.

Team
PROJECT REVIEW

각 개인 기능구현 소개.



회원관리 구현

노유리

Membership

- join membership
- log in
- show/edit Information
- delete information

Join Member

```java
// 회원가입
public void joinMember() {


    Util.keyboard.nextLine();
    System.out.println("회원가입 양식에 따라 입력해주세요!");
    System.out.print("I      D :");
    String id = Util.keyboard.nextLine();

    while (loco.bit.containsKey(id)) {
        System.out.println("ID가 사용중입니다.다시 입력해주세요.");
        System.out.print("I      D :");
        id = Util.keyboard.nextLine();
    }
    System.out.print("Password :");
    String pw = Util.keyboard.nextLine();

    System.out.print("N A  M E :");
    String name = Util.keyboard.nextLine();

    System.out.print("PhoneNum :");
    String pNum = Util.keyboard.nextLine();

    m = new MemberInfo(id, pw, name, pNum);
    System.out.println(id+"님, "+"회원가입 되었습니다. ");
    mList.add(m);

    loco.bit.put(id, pw);
}
```

Show Information

```java
// 회원정보 출력
public void showAllData() {
    // int index = searchIndex(id);
    String id = null;
    if (mList.size() > 0) {
        for (int i = 0; i < mList.size(); i++) {
            mList.get(i).showData();
            System.out.println("============================");
            break;
        }
    } else {
        System.out.println("등록된 회원정보가 없습니다.");
    }
}
```

Edit Information

```java
// 회원정보 수정
public void editInfo() {
    System.out.print("INSERT-ID: ");
    String id = Util.keyboard.nextLine();

    while (searchIndex(id) < 0) {
        System.out.println("아이디가 일치하지 않습니다.");
        System.out.print("I      D :");
        id = Util.keyboard.nextLine();
    }

    System.out.println("["+id+"님의 정보를 수정합니다]");

    System.out.print("PASSWORD :");
    String pw = Util.keyboard.nextLine();

    System.out.print("N  A  M E:");
    String name = Util.keyboard.nextLine();

    System.out.print("Phone Num:");
    String pNum = Util.keyboard.nextLine();

    m = new MemberInfo(id, pw, name, pNum);
    int index = searchIndex(id);
    mList.add(index, m);
    loco.bit.put(id, pw);
    System.out.println(id + "님의 정보가 정상적으로 수정되었습니다.");
}
```

Delete Information

```java
// 회원정보 삭제
public void deleteInfo() {
    System.out.println("insert ID ");
    String id = Util.keyboard.nextLine();
    while (!(bit.containsKey(id))) {
        System.out.println("wrong ID!");
        System.out.print("I      D :");
        id = Util.keyboard.nextLine();
    }
    System.out.print("PASSWORD :");
    String pw = Util.keyboard.nextLine();
    while (bit.containsKey(id) && !(bit.get(id).equals(pw))) {
        System.out.println("wrong ID!" + "\nPASSWORD :");
        pw = Util.keyboard.nextLine();
    }

    if (bit.get(id) != null && bit.get(id).equals(pw)) {
        mList.remove(searchIndex(id));
        bit.remove(id, pw);
        System.out.println("회원 정보가 확인되었습니다.");
        System.out.println("---------------------------");
    }
    System.out.println("정보가 정상적으로 삭제되었습니다.");
    return;
}
```

Point of improvement

array list/hash map 적용 과정에 정보가
정상적으로 저장/수정되지 않는 오류 발생

✓ membership point 관리/coupon 적립 등을
적용하고 싶었지만..

Log-in 구현

최아리

로그인 구현 (1) _ 메서드 호출

```
public void login() {
    System.out.println("---------------------------");
    System.out.println("1.Home\n2.Login");
    System.out.print("Select Number:");
    select = keyboard.nextInt();
    keyboard.nextLine();
    if(select==1) {
        return;
    }else {

        choice:while(true) {

            System.out.println("=====  L  O  G  I  N  =====");
            System.out.print("ID: ");
            String id = keyboard.nextLine();

            System.out.print("Password: ");
            String pw = keyboard.nextLine();

            if(bitloco.containsKey(id)) {
                if(bitloco.get(id).equals(pw)) {
                    System.out.println(id+"님,로그인 되었습니다.");
                    return;
                }else {
                    System.out.println("비밀번호가 일치하지 않습니다.");
                }
            }else {
                System.out.println("존재하지 않는 회원입니다.다시 확인해주세요.");
            }
        }
    }
}
```

컬렉션프레임워크를 이용한 로그인 구현

HashMap <K , V>

```
public void login() {
    System.out.println("---------------------------");
    System.out.println("1.Home\n2.Login");
    System.out.print("Select Number:");
    select = keyboard.nextInt();
    keyboard.nextLine();
    if(select==1) {
        return;
    }else {

        choice:while(true) {

            System.out.println("=====  L  O  G  I  N  =====");
            System.out.print("ID: ");
            String id = keyboard.nextLine();

            System.out.print("Password: ");
            String pw = keyboard.nextLine();

            if(bitloco.containsKey(id)) {
                if(bitloco.get(id).equals(pw)) {
                    System.out.println(id+"님,로그인 되었습니다.");
                    return;
                }else {
                    System.out.println("비밀번호가 일치하지 않습니다.");
                }
            }else {
                System.out.println("존재하지 않는 회원입니다.다시 확인해주세요.");
            }
        }
    }
}
```

로그인 구현 (2) _ 실제로 구현 되는곳

```
1. M E M B E R S
2. J O I N
3. M E N U
4. Q U I T
============================
Select Number:1
1. H O M E
2. L O G I N
Select Number:2
■■■■■  L  O  G  I  N  ■■■■■
I  D:1
P  W:1
*****  접속시간을 체크합니다 ******
[존재하지 않는 회원입니다.다시 확인해주세요.]
회원가입하시겠습니까?
1. J O I N
2. H O M E
Select Number:
```

```
   12시 54분 17초 종료!
Thank you for using.
Please come again next time.
                    _ BITLOCO
```

```
■■■■■  L  O  G  I  N  ■■■■■
I  D:1
P  W:1
*****  접속시간을 체크합니다 ******
 12시 53분 28초 접속!
[ 1, Signed in! ]
============================
```

1. 가입된 멤버가 로그인 후 회원 전용 메뉴를 이용
2. SimpleDateFormat을 이용하여 접속,종료시간 표시

음료 메뉴

유동준

Beverage Menu (음료 메뉴)

```java
public interface Menu_Inter {

    void showPrint();

    // 메인 메뉴 선택창
    public int MEMBERS=1, JOIN=2, MENU=3,QUIT=4;


    //메뉴 선택 시 쓰는 것들
    // 음료 & 푸드 5에 fruit추가 6부터 shift
    public int AMERICANO = 1, CAPPUCHINO =2, SPARKLING =3, LEMONADE=4, FRUITJUICE = 5
    ,CHEEZE = 6, CHOCO = 7, COOCKIES = 8, SANDWITCHES =9;


    //음료인가 푸드인가
    public int BEVERAGE = 1, FOOD = 2;

    //음식은 1=안땡긴가 음료1=차가운음료
    // 바나나 딸기 딸바 항목 추가
    int FOOD_COLD = 1,FOOD_HOT = 2, BEV_COLD = 1, BEV_HOT=2, BEV_BANANA=1, BEV_STRAWBERRY=2, BEV_CHOCO=3;

}
```

```java
abstract class Beverage extends Menu {

    private String size;
    private String cold;
    private String fruit;
    private String sugar;
    private String cup;

    /* 생성자 */
    public Beverage() {
        super();
        this.size = "small";
        this.cold = "iced";
        this.fruit = "";
        this.sugar = "";
        this.cup = "";
    }

    // 사이즈업하는 메서드
    public void sizeUP() {
        size = "large";
        setPrice(getPrice() + 1000);
    }

    // hot으로 바꿔주는 메서드
    public void noCold() {
        cold = "hot";
    }

    // 바나나를 위한 메소드
    public void banana() {
    }

    public String getSize() {
        return this.size;
    }

    public String getCold() {
        return this.cold;
    }

    public String getFruit() {
        return this.fruit;
    }

    public String getSugar() {
        return this.sugar;
    }

    public String getCup() {
        return this.cup;
    }


    public abstract void showPrint();

    @Override
    public void showProduct() {
        System.out.println(getSugar() + " " +
getCold() + " " + getFruit() + " " + getName
+ " " + getSize() + " | " + getPrice() + " 원
+ getCup()+" ");
    }
```

Beverage Menu (음료 메뉴)

```java
//레몬에이드 클래스
class Lemonade extends Beverage {

    // 생성자
    public Lemonade() {
        super();
        setPrice(4000); // 상품의 가격 정해줌.
        setName("레몬에이드"); // 상품의 이름 정해줌(오버라이드)
    }

    @Override
    public void showPrint() {
        System.out.println(getName() + " | " + getPrice() + " 원");
    }
}

//과일쥬스 클래스
class FruitJuice extends Beverage {

    // 생성자
    public FruitJuice() {
        super();
        setPrice(4000);
        setName("과일 쥬스");
    }

    @Override
    public void showPrint() {
        System.out.println(getName() + "   | " + getPrice() + " 원 (바나나,딸바,바)");
    }
```

```java
//아메리카노 클래스 AMERICANO = 1
class Americano extends Beverage {

    // 생성자
    public Americano() {
        super();
        setPrice(4000); // 상품의 가격 정해줌.
        setName("아메리카노"); // 상품의 이름 정해줌(오버라이드)
    }

    @Override
    public void showPrint() {
        System.out.println(getName() + " | " + getPrice() + " 원");
    }
}

//카푸치노 클래스 CAPPUCHINO =2, SPARKLING =3, LEMONADE=4;
class Cappuchino extends Beverage {

    // 생성자
    public Cappuchino() {
        super();
        setPrice(5000); // 상품의 가격 정해줌.
        setName("카푸치노"); // 상품의 이름 정해줌(오버라이드)
    }

    @Override
    public void showPrint() {
        System.out.println(getName() + "    | " + getPrice() + " 원");
    }

}
```

Beverage Menu (음료 메뉴)

MenuSelect 내용 추가

```
// 딸기와 바나나를 선택
if (b.get(b.size() - 1) instanceof FruitJuice) {
    System.out.printf("%d.바나나 %d.딸바 %d.초바 \n", Menu_Inter.BEV_BANANA, Menu_Inter.BEV_STRAWBERRY,
            Menu_Inter.BEV_CHOCO);

    int fruit = Util.keyboard.nextInt();

    if (fruit >=  Menu_Inter.BEV_BANANA && fruit <=  Menu_Inter.BEV_CHOCO) {

        if (fruit == Menu_Inter.BEV_BANANA) {
        orderBev.get(index).banana();

        } else if (fruit == Menu_Inter.BEV_STRAWBERRY) {
        orderBev.get(index).strawberry();

        } else if (fruit == Menu_Inter.BEV_CHOCO) {
        orderBev.get(index).choco();

        } else {
        System.out.println("!!다시 입력!!");
        fruit = Util.keyboard.nextInt();
        }

            System.out.println("설탕을 넣으시겠습니까? 1.네 2.아니요");

            int sugarFree = Util.keyboard.nextInt();

            if (sugarFree == 2) {
            orderBev.get(index).sugarFree();
            }
        }
    }
```

Beverage Menu (음료 메뉴)

MenuSelect 내용 추가

```
}order.add(orderBev.get(index));

// 개인 컵을 사용하면 -500원 할인
System.out.println("개인 컵을 사용하시면 500원 할인됩니다. 개인컵을 사용하시겠습니까? 1.예 2.아니오");

int selfCup = Util.keyboard.nextInt();

if(selfCup ==1) {
    orderBev.get(index).selfCup();
}

while(selfCup>2) {
    System.out.println("다시 입력해주세요");

    selfCup = Util.keyboard.nextInt();

}
```

추가, 변경, 수정 에서의 문제

- 아주 작은 변경에도 잦은 오류
- 같은 내용이 중복 출력되는 경우
- 배열의 size가 계속 오류로 인하여 장바구니가 주문없이 초과
- 논리적인 오류 등등,,,

Beverage Menu (음료 메뉴)

추가, 변경, 수정 에서의 문제

- 아주 작은 변경에도 잦은 오류
- 같은 내용이 중복 출력되는 경우
- 배열의 size가 계속 오류로 인하여 장바구니가 주문없이 초과
- 논리적인 오류 등등,,,

서로 서로 연결되어 있는 코드 전부를 변경하는것에 잦은 실수

추가, 수정의 잦은 반복에 코드들이 섞이고 난잡해짐

객체가 추가 될 때 마다 배열의 범위가 변함

결론!! *구조, 설계에 대한 이해도 부족*

-- 확실한 이해를 시작으로 작업해야 더 효율적으로 대처할 수 있다.

푸드메뉴

소종원

인터페이스

```java
package util;

public interface Menu_Inter {

    void showPrint();

    // 메인 메뉴 선택창
    public int MEMBERS = 1, JOIN = 2, MENU = 3, QUIT = 4;

    // 메뉴 선택 시 쓰는 것들
    // 음료 & 푸드
    public int AMERICANO = 1, CAPPUCHINO = 2, SPARKLING = 3, LEMONADE = 4, FRUITJUICE = 5, CHEEZE = 6, CHOCO = 7,
            COOCKIES = 8, SANDWITCHES = 9, SCONE = 10, HONEY_BREAD = 11;

    // 음료인가 푸드인가
    public int BEVERAGE = 1, FOOD = 2;

    // 음식은 1=안뎁힌거 음료1=차가운음료
    // 바나나 딸기 딸바 항목 추가
    int FOOD_COLD = 1, FOOD_HOT = 2, BEV_COLD = 1, BEV_HOT = 2, BEV_BANANA = 1, BEV_STRAWBERRY = 2, BEV_CHOCO = 3;

}
```

프로젝트 구조상 가장 상위에 위치하게 되는 인터페이스와 그 안에 상수로 지정된 변수들.

인터페이스를 구현한 추상클래스들

```java
abstract class Food extends Menu {

    /*
     * 변수선언
     *
     * @hot: 기본이 따뜻X, 1로 초기화 (따뜻하면 2로) // 베이커리류만 오버라이딩하면 됨.
     *
     * @honey: 기본이 정량, 2가 생크림 추가
     */
    private String hot;
    private String cream;
    private String sinamon;
    private String honey;

    public String getCream() {
        return cream;
    }

    public void setCream(String cream) {
        this.cream = cream;
    }

    public String getSinamon() {
        return sinamon;
    }

    public void setSinamon(String sinamon) {
        this.sinamon = sinamon;
    }

    public String getHoney() {
        return honey;
    }
```

Menu 추상 클래스를 상속받은 Food 클래스

```java
public abstract class Menu implements Menu_Inter{


    /* 실제 상품 클래스에서 count 추가해주는 메서드 있어야 함.
    private int price;
    private int cnt;
    private String name;

    /* 생성자 */
    public Menu() {
    }

    public abstract void showPrint();

    public void showProduct() { //주문선택한게 맞는지 물어보는 메서드
        System.out.println(getName() + " | " + getPrice() + " 원");
    }
```

인터페이스를 구현한 Menu 추상 클래스

인터페이스에서 이어 받은 추상 메소드

추상클래스를 상속받은 하위 클래스

```
class Cheeze extends Food {

    public Cheeze() {

        super();
        this.setPrice(5500);
        this.setName("치즈케이크");
    }

    @Override
    public void showPrint() {
        System.out.println(this.getName() + " | " + this.getPrice() + " 원");

    }

}

class Choco extends Food {

    public Choco() {

        super();
        setPrice(6000);
        setName("초코케이크");
    }

    @Override
    public void showPrint() {
        System.out.println(getName() + " | " + getPrice() + " 원");
    }

}
```

Food 클래스를 상속받은 하위 클래스들과 인터페이스에서 타고 내려온 메소드를 오버라이딩

```
public void addCream() {

    setName(getName() + "(생크림 추가)");
    setPrice(getPrice() + 800);
}

public void addSinamon() {

    setName(getName() + "(시나몬 추가)");
    setPrice(getPrice() + 500);
}

public void addHoney() {

    setName(getName() + "(꿀 추가)");
    setPrice(getPrice() + 600);
}

@Override
public void showProduct() {
    System.out.println(getHot() + " " + getName() + " | " + getPrice() + " 원");
}
```

Food 클래스에서 작성한 메소드들로 상속받은 하위클래스는 이 메소드들을 모두 사용 가능

하위 클래스와 메소드의 활용

```
// 푸드 커스텀
void customFood(ArrayList<Food> f) {

    // @index: 지금 커스텀하고있는 메뉴의 인덱스값
    int index = orderFood.size() - 1;

    System.out.println("선택하신 푸드: " + orderFood.get(index).getName());

    if ((f.get(f.size() - 1) instanceof Cookies || f.get(f.size() - 1) instanceof Sandwiches)
            || f.get(f.size() - 1) instanceof Scone || f.get(f.size() - 1) instanceof Honeybread
            || f.get(f.size() - 1) instanceof Cheeze || f.get(f.size() - 1) instanceof Choco) {

        System.out.printf("%d.기본 %d.따뜻하게\n", Menu_Inter.FOOD_COLD, Menu_Inter.FOOD_HOT);

        int answer = Util.keyboard.nextInt();

        while (answer < Menu_Inter.FOOD_COLD || answer > Menu_Inter.FOOD_HOT) {
            System.out.println("!!다시 입력!!");
            answer = Util.keyboard.nextInt();

        }

        if (answer == Menu_Inter.FOOD_HOT) {
            orderFood.get(index).noCold();
        }

        System.out.println("생크림 추가. (800 원)\n1.예 2.아니오");
        int creamUp = Util.keyboard.nextInt();

        if (creamUp == 1) {
            orderFood.get(index).addCream();
        }
```

인터페이스에서 지정한 상수들과 인터페이스를 이어 받아 내려와 상속받은 클래스들의 메소드들을 이용해 메소드 작성

주문Sys 구현

ArrayList와 FileWriter 중심으로

김나연

메뉴판 보기 –메뉴판 보여주기 + 현재 수량

구현내용

로그인 후 > ORDER 선택 > 원하는 메뉴선택 >

ArrayList<Menu> menu; // 메뉴판보여주기

* 생성자와 메서드 호출을 통해 주문가능수량 setting

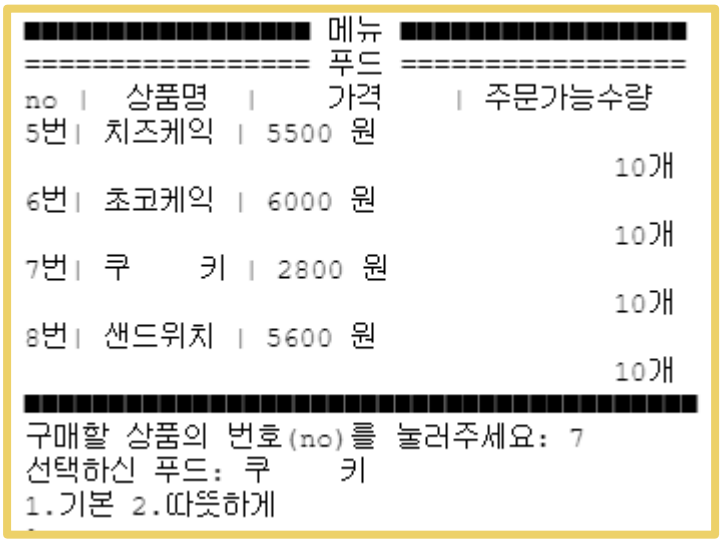
```
==========================================
        어떤 메뉴를 원하십니까?
1.음료 2.푸드 3.주문보기 4.결제 5.나가기
==========================================
```

상품 선택 –현재 주문 중인 상품 커스터마이징하기

구현내용

음료: ICED/HOT && SIZEUP
푸드: 쿠키/샌드위치만 HOT or NOT?
커스텀: If / Switch 문으로 구현

초반 간단하였으나 커스텀 종류가 많아져 refactoring issue 발생.

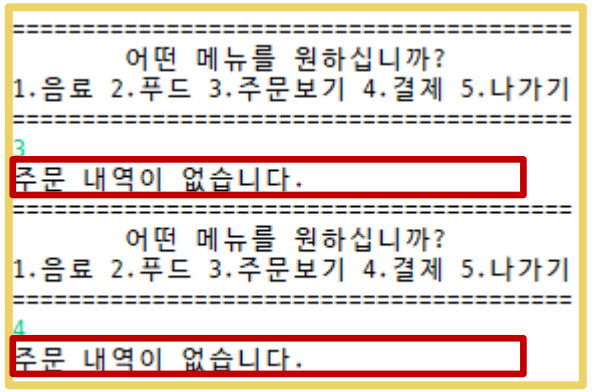


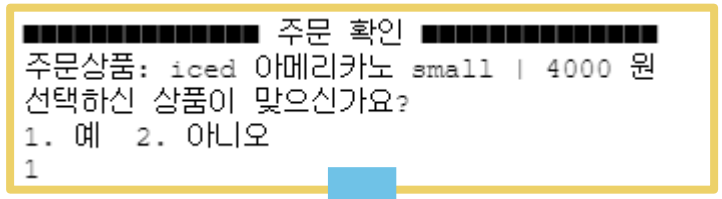
주문확정 –임시객체에서 주문이 확정된 객체로 push

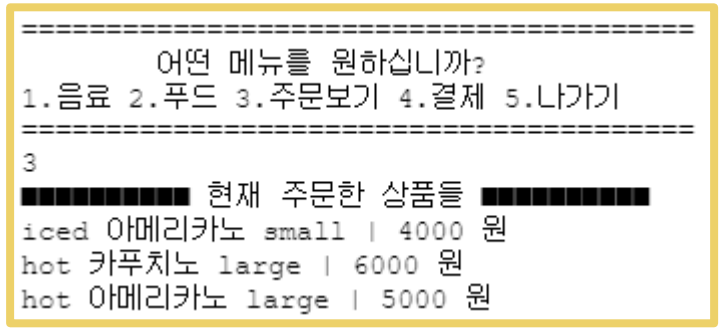
구현내용

ArrayList<Beverage> orderBev; //주문-음료(임시)
ArrayList<Food> orderFood; // 주문-푸드(임시)

>영수증에 넣기 위해 임시로 만듬.







영수증 format으로 저장, 출력

구현내용

ArrayList<Menu> order; // 영수증 출력, file 출력

주문시간 및 상품 영수증 form으로 출력
-BufferedWriter, FileWriter 사용
-Bill.txt 파일로 저장.

* String 배열을 이용, File에 출력할 주문내용 구현

```
========================================
        어떤 메뉴를 원하십니까?
1.음료 2.푸드 3.주문보기 4.결제 5.나가기
========================================
4
....결제창으로 넘어갑니다....

■■■■■■■■■■■■ B I T L O C O ■■■■■■■■■■■■
대표자: 최아리
주문시각: 2019/05/29 12:33:21
----------------------------------------
================ 메뉴 ================
   상  품  명          ¦    가   격
----------------------------------------
 hot  아메리카노 small ¦ 4000 원
----------------------------------------
 iced  카푸치노 large ¦ 5500 원
----------------------------------------
기본 허니브레드(시나몬 추가)(꿀 추가) ¦ 7
----------------------------------------
기본 치즈케이크(생크림 추가)(시나몬 추가)
----------------------------------------
기본 쿠키       ¦ 2800 원
----------------------------------------┼--
 T O T A L :        W          26800
========================================
주문이 완료되었습니다.
========================================
        어떤 메뉴를 원하십니까?
1.음료 2.푸드 3.주문보기 4.결제 5.나가기
========================================
```

아쉬운 점과 수정사항

- Refactoring이 어려웠음. 주문(임시)객체를 영수증으로 인풋 시null-point-exception 발생.
  - 발전방향: 처음 기획단계부터 기능에 맞는 class 제작.
  - 다른 사람들의 이해도를 위한 코딩 생각해 볼 것.
- 예외상황을 많이 생각하여 메서드가 길어짐, 원하는 곳으로 리턴 throw 어려웠음.
- 추가 문제점: 리팩토링 단계에서 메뉴판에 표시된 수량cnt가 연동되지 않았음.
  - 해결방향: Map을 써볼 것. Value에 수량을 입력하고 연동처리 해 볼 예정.

EVENT 구현

송주은

EventMenu(RandomCoffee)

```
public void randomCoffee() throws IOException {
        System.out.println("오늘의 음료를 추천하고 있는 중입니다.");
        try {
                Thread.sleep(2000);
                int choice = (int) (Math.random() * 4 + 1);
                custum(choice);
                showCurOrder();
                checkOrder(choice);

                } catch (InterruptedException e) {
                        e.printStackTrace();
                }
}
```

EventMenu(RandomFood)

```
public void randomFood() throws IOException {
        System.out.println("오늘의 음료를 추천하고 있는 중입니다.");
        try {
                Thread.sleep(2000);
                int choice = (int) (Math.random() * 6 + 5);
                custum(choice);
                showCurOrder();
                checkOrder(choice);

                } catch (InterruptedException e) {
                        e.printStackTrace();
                }
}
```

EventMenu(customBev)

```
@Override
        void customBev(ArrayList<Beverage> b) {

                int index = orderBev.size() - 1;

                System.out.println("오늘의 추천 음료 - " + orderBev.get(index).getName());

                // coffee 류면 hot/iced 선택
                if (b.get(b.size() - 1) instanceof Americano || b.get(b.size() - 1) instanceof
Cappuchino) {
                        System.out.printf("%d.차갑게 %d.따뜻하게₩n", Menu_Inter.BEV_COLD,
Menu_Inter.BEV_HOT);

                        int answer = Util.keyboard.nextInt();
```

EventMenu(customBev)

```
                while (answer < Menu_Inter.BEV_COLD || answer > Menu_Inter.BEV_HOT)
{
                        System.out.println("!!다시 입력!!");
                        answer = Util.keyboard.nextInt();
                }
                if (answer == Menu_Inter.BEV_HOT) {
                        orderBev.get(index).noCold();
                }
        } // hot<>iced change
```

EventMenu(customBev)

```
            System.out.println("사이즈업하시겠습니까? 1.예 2.아니오");
            int sizeUp = Util.keyboard.nextInt();

            if (sizeUp == 1) {
                        orderBev.get(index).sizeUP();
            }

            while (sizeUp > 2) {
                        System.out.println("다시 입력해주세요.");
                        sizeUp = Util.keyboard.nextInt();
            }
            order.add(orderBev.get(index));
}
```

EventMenu(customFood)

```
void customFood(ArrayList<Food> f) {
                    // TODO Auto-generated method stub
                    int index = orderFood.size() - 1;

                    System.out.println("오늘의 추천 FOOD - " + orderFood.get(index).getName());

                    // 샌드위치/쿠키만
                   if (f.get(f.size() - 1) instanceof Cookies || f.get(f.size() - 1) instanceof Sandwiches)
{
                              System.out.printf("%d.기본 %d.따뜻하게₩n", Menu_Inter.FOOD_COLD,
Menu_Inter.FOOD_HOT);

                              int answer = Util.keyboard.nextInt();
```

EventMenu(customFood)

```
                        while (answer < Menu_Inter.FOOD_COLD || answer >
Menu_Inter.FOOD_HOT) {
                                System.out.println("!!다시 입력!!");
                                answer = Util.keyboard.nextInt();

                        }

                        if (answer == Menu_Inter.FOOD_HOT) {
                                orderFood.get(index).noCold();
                        }
                }
                order.add(orderFood.get(index));
        }
```

EventMenu(checkOrder)

```
public void checkOrder(int select) throws IOException {
// TODO Auto-generated method stub
System.out.println("■■■■■■■■■■■■■■■ 주문 확인 ■■■■■■■■■■■■■■■ ");
System.out.print("주문상품: ");

order.get(order.size() - 1).showProduct();

System.out.println("선택하신 상품이 맞으신가요?");
System.out.println("1. 예  2. 아니오");

// 상품재확인 1.맞다. 2.아니다(== order에서 빼라.)
int choice = Util.keyboard.nextInt();
```

EventMenu(checkOrder)

```
while (choice > 2 || choice < 1) {
System.out.println("!!다시 입력!!");
choice = Util.keyboard.nextInt();
}

if (choice == 2) {
order.remove(order.size() - 1);
System.out.println("메뉴창으로 다시 돌아갑니다.");
return;

}
```

EventMenu(checkOrder)

```
else if (choice == 1) {

int curCnt = menu.get(select - 1).getCnt();
menu.get(select - 1).setCnt(--curCnt);
showBill(id);
}

}
```

Event (ShowBill)

```
@Override
        void showBill(String id) {
                System.out.println("결제 하시겠습니까?");
                System.out.println("1. 예₩t2. 아니오");

                int choice = Util.keyboard.nextInt();
                if (!(choice < 1 || choice > 2)){
                        if(choice == 1){
                                if (order.size() != 0) {
                                        System.out.println("....결제창으로
넘어갑니다....₩n");
```

Event (ShowBill)

```
try{
                billFormat(id);
                m.getOrderTime();
                System.out.println("주문이 완료되었습니다.");
                }catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                }else {
                        System.out.println("주문 내역이 없습니다.");
        }
```

Event (ShowBill)

```
}else {
                        System.out.println("1 혹은 2만 입력해주세요.");
                        return;
                }
                }//if문 종료

                }


        }
```

Event (Show Last order)

```
void showlastOrder() throws IOException{
File file = new File("C:₩₩Users₩₩5CLASS-184₩₩Documents₩₩javaClass₩₩Cafe₩₩bill.txt");
                    FileReader fileReader;
                    try {
                                fileReader = new FileReader(file);
                                BufferedReader bReader = new BufferedReader(fileReader);
                                String line="";
                                while((line = bReader.readLine()) != null) {
                                            System.out.println(line);
                                }bReader.close();
                    } catch (FileNotFoundException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
```

EventMenu

```
void showEvent() throws IOException {

                System.out.println("이벤트 창입니다.");

                while (true) {
                        System.out.printf("어떤 이벤트를 원하십니까?
₩n%d.랜덤커피₩t %d.랜덤FOOD₩t %d.직전 선택메뉴 구매₩t %d. 나가기₩n", 1, 2, 3, 4);

                        int choice = Util.keyboard.nextInt();
                        switch (choice) {
                        case 1:
                                randomCoffee();  //랜덤커피 실행하는 메서드
                                break;
```

EventMenu

```
                case 2:
                    randomFood();    //랜덤푸드 실행하는 메서드
                    break;


                case 3:
                    showlastOrder();  //직전 선택메뉴 구매를 실행하는 메서드
                    break;

                case 4:
                    return;
                }

            }
        }
```

EventMenu 구현하고 싶었던 것

3번째 이벤트인 직전 내역 결제,
읽어온 내용에서 특정 키워드(메뉴이름)을 찾아
추출 후 바로 결제로 이동을 구현

음료+푸드 set 추천

Cafe. BITLOCO

THANK YOU !